Training Parameterized Quantum Circuits with Triplet Loss

Christof Wendenius[0000−0002−3289−8690], Eileen Kuehn[✉][0000−0002−8034−8837], and Achim Streit[0000−0002−5065−469X]

Karlsruhe Institute of Technology, Hermann-von-Helmholtz Platz 1, 76344 Eggenstein-Leopoldshafen, Germany
{christof.wendenius,eileen.kuehn,achim.streit}@kit.edu

Abstract. Training parameterized quantum circuits (PQCs) is a growing research area that has received a boost from the emergence of new hybrid quantum classical algorithms and Quantum Machine Learning (QML) to leverage the power of today's quantum computers. However, a universal pipeline that guarantees good learning behavior has not yet been found, due to several challenges. These include in particular the low number of qubits and their susceptibility to noise but also the vanishing of gradients during training. In this work, we apply and evaluate Triplet Loss in a QML training pipeline utilizing a PQC for the first time. We perform extensive experiments for the Triplet Loss based setup and training on two common datasets, the MNIST and moon dataset. Without significant fine-tuning of training parameters and circuit layout, our proposed approach achieves competitive results to a regular training. Additionally, the variance and the absolute values of gradients are significantly better compared to training a PQC without Triplet Loss. The usage of metric learning proves to be suitable for QML and its high dimensional space as it is not as restrictive as learning on hard labels. Our results indicate that metric learning provides benefits to mitigate the so-called barren plateaus.

Keywords: Metric learning · Quantum Machine Learning · Parameterized Quantum Circuits.

1 Introduction

In recent years, Quantum Machine Learning (QML) has become a highly active and promising field of research starting to leverage the enormous potential of Quantum Computers (QC) [7]. Current devices of the noisy intermediate-scale quantum (NISQ) era are still error-prone and unable to process large-scale datasets due to the coupling between algorithm complexity and noise [22]. However, through tremendous efforts, multiple advances have been made in both theory and practice to use QCs in and for machine learning [26,28].

Parameterized quantum circuits (PQCs), also referred to as Quantum Neural Networks (QNNs), are one of the most studied aspects of QML as their structure and trainable parameters are reminiscent of neural networks [18]. Hoping to

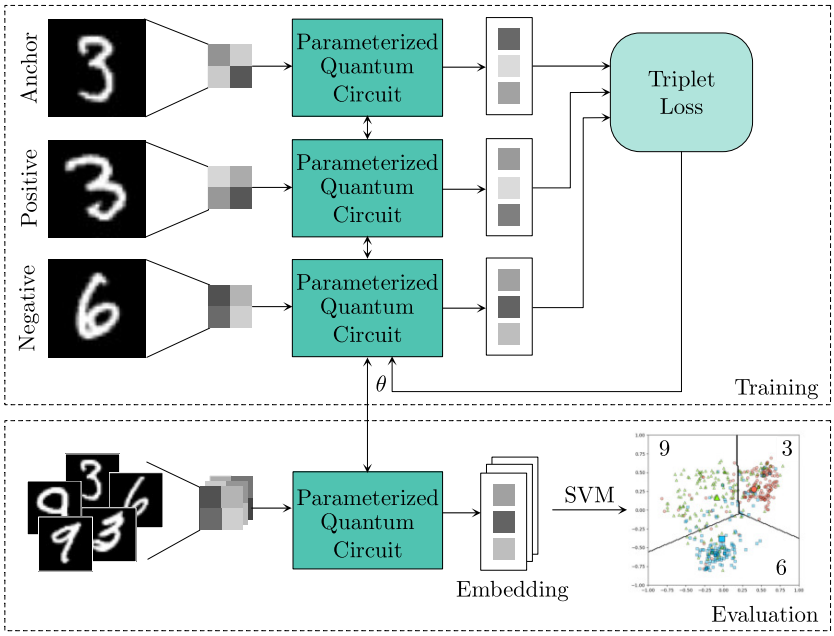


Fig. 1: Overview of our proposed Triplet Loss approach and the two pipelines for training and evaluation. A parameterized quantum circuit is used to create an embedding of the inputs for anchor, positive, and negative that are required for Triplet Loss. During training, these embeddings are taken to minimize the Triplet Loss. In the evaluation, the embeddings of the training set are used for training a linear SVM classifier. The accuracy of the trained quantum model is evaluated based on the embeddings of a dedicated test set.

achieve similar leaps in machine learning tasks as their classical counterparts, particular attention has been focused on training PQCs. So-called barren plateaus, where the gradients vanish towards zero with an increasing number of qubits, often hinder successful training [17]. Although advances have been made towards mitigating these effects [5,9,26], there is still no universal answer to guarantee good learning behavior. Despite many proposed methods, effectively using classical data on QCs remains a challenge as well. Since PQCs are part of a hybrid quantum-classical training loop, the embedding of features into the quantum model and the interpretation of measurements for the loss function have to be studied further.

In this work, we propose to approach training PQCs from a new angle by applying a ranking loss function from the task of metric learning. These loss functions do not aim to get a model output that corresponds to a label for classification but optimize the relative distances between different inputs in the embedding space of the model [13]. Intuitively, we expect this to be a viable application of quantum computers for two reasons: On the one hand, entanglement of qubits allows to represent a huge embedding space internally, quadratic in the

number of qubits. On the other hand, the measurement used to transition from quantum to classical state corresponds to a scalar product between quantum states, suitable as a native metric computation.

In particular, we propose to apply the so-called *Triplet Loss* which has been used to great effect for the training of classical neural networks [23]. This choice is somewhat arbitrary with respect to similar loss functions from metric learning such as contrastive loss. While we expect triplet loss to better populate the embedding space, a thorough evaluation of different metric loss functions is out of scope for this paper.

For triplet loss, as for any ranking loss function, similar inputs, i.e. inputs with the same class, should generate model outputs in proximity to each other. Analogously, non-matching inputs should be further apart in the embedding space. In this way, the model does not need to create a specific output for each input but can make use of the entire embedding space. By transferring this kind of loss function and training scheme onto quantum devices, training of PQCs is more unrestricted and can make use of the full range of the underlying computational space, the so-called expressibility.

To demonstrate the applicability of Triplet Loss for training of PQCs, we run several experiments on the moon [1] and MNIST [15] datasets. For this purpose, we implement two pipelines for training and evaluating our approach, which can be seen in Fig. 1. During training, the Triplet Loss is used to optimize the embeddings calculated by using a PQC by maximizing distances between dissimilar samples and minimizing distances for similar samples. With our work, we aim to show that Triplet Loss can achieve comparable results to training as it is currently common in the literature. In addition, higher gradient values indicate that the approach could achieve better learning behavior with larger circuits on NISQ devices. While metric learning is not new in the area of quantum computing [16,27], it has neither been used in the regime of PQCs nor has it been evaluated with respect to variances of gradients during training.

The paper is organized as follows. In Section 2 we give a brief overview on QNN and QML as well as the training of PQCs and discuss related work. This is followed by the basics on metric learning and Triplet Loss in Section 3, which are necessary to understand our approach in Section 4. Our experiments, datasets, and results are presented in detail in Section 5 We conclude in Section 6.

2 Quantum Neural Networks and Quantum Machine Learning

Quantum Neural Networks (QNNs) can be thought of as a generalization of Deep Neural Networks (DNNs). While in both cases a classical optimizer updates the models parameters θ to minimize a predefined loss function $\mathcal{L}$, the main difference lies in the model to be trained, as illustrated in Fig. 2. In the case of QML, the model is based on the principles of quantum mechanics such as superposition, entanglement and interference. In practice, there are numerous realizations of QNN models but most architectures share the same structure:

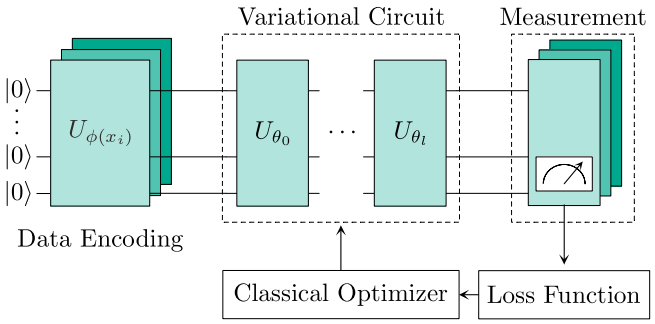


Fig. 2: Common quantum machine learning pipeline to train a parameterized quantum circuit. The parameterized quantum circuit is composed of three parts: the data encoding $U_{\phi(x)}$, variational circuit U_θ and measurement. The variational circuit has up to l layers. The training runs on n qubits. Measurement results are fed into a classical loss function that is used by a classical optimizer for minimization and update of θ. The QML pipeline is denoted as hybrid as a quantum and classical part are jointly used to harness the power of quantum computers.

Parameterized Quantum Circuits [2,4]. The hope is that with QNNs the power of quantum computers can be harnessed to outperform DNNs for specific use cases or quantum data.

A circuit contains any number of gates that are assigned to one or more qubits. Gates can transform the quantum state of each qubit they are assigned to corresponding to computational steps on a classical computer. In the visual representation of a circuit, the individual qubits are shown as a horizontal line. The structure of the circuit and thus the arrangement and composition of the gates is often referred to as an ansatz.

A PQC constitutes three parts: an encoder circuit $U_{\phi(x)}$ parameterized by the vector $\phi(x)$, a variational circuit U_θ parameterized by a vector θ, and the measurement part M. In this case $\phi(\cdot)$ relates to a preprocessing scheme that maps the input data to a vector that can be applied on the circuit. For each of these parts different gates can be chosen to describe transformations of the quantum state. While the selection, composition, and evaluation of a proper ansatz is not part of this paper, this is still a highly relevant field of research [3,4] impacting the trainability and practical usefulness of PQCs.

Purpose of the encoding part is loading classical data by encoding it into the quantum state of the qubits. In literature, there are many data encoding methods [14,19] and research is still ongoing as the data encoding itself is relevant to enable high expressibility of PQCs [24]. Some well-known approaches are for example (i) basis encoding, (ii) angle encoding, or (iii) amplitude encoding. For further details regarding data encoding as well as theoretical aspects of quantum computing we refer the interested reader to the work by Nielsen and Chuang [19].

Once the classical data sample is prepared, the variational circuit U_θ is applied to this state. This trainable part U_θ consists of trainable single-qubit quantum gates and fixed two-qubit quantum gates. Here, the weights known from classical NNs are in the form of rotational parameters. As with the encoding part, the variational circuit can be composed in different ways where using more gates enables higher expressibility but also challenged trainability [11]. One important concept that has established in literature is the hardware-efficient ansatz that introduces a layered structure where the placement of gates is equal in each layer l but parameterized by an own set of trainable parameters θ_l. The variational circuit U_θ is the sum of all its layers, that is $U_\theta = \prod_{l=1}^{L} U_{\theta_l}$ where L is the number of layers.

This processing step is followed by the quantum measurement to extract quantum information into its classical form. During measurement of a single qubit, its quantum state collapses to one classical single value, 0 or 1. Therefore, most QML approaches require a number of shots for measurements to estimate the expectation value with high probability, e.g. $1,000$. The extracted information can either be used directly as a predicted label e.g. in case of quantum classifiers [14,27] or a hidden feature depending on the appropriate QML pipeline.

The whole QNN pipeline is commonly trained and executed in a hybrid fashion to accommodate for current NISQ devices while still being able to explore the potential advantages such as speed-ups in training and processing. While the PQC is purely based on principles of quantum mechanics, other parts including optimization are primarily done in a classical fashion. Much development in combining quantum and classical models has already been done by the community, but systematic research on the trade-offs and potential advantages is still ongoing. Furthermore, especially the generalization error of QML is of interest that directly depends on the training error. Thus, our approach complements these studies by evaluating the use of Triplet Loss in the QNN training pipeline.

2.1 Related Work on Training Parameterized Quantum Circuits

The trainability of PQCs is paramount to advance the field of QML. However, as mentioned before the expressibility of a quantum circuit challenges its trainability and comes along with the effect of exponentially vanishing gradients in the number of qubits, as a function of the number of layers, the so-called barren plateaus [17].

Different strategies have been proposed to mitigate barren plateaus, e.g. by initializing the parameters of variational circuits in form of identity blocks [9], or by pre-training a variational circuit based on layerwise learning strategies [26] as has been shown to provide good results in classical machine learning [10]. However, most of these strategies do not strictly guarantee to prevent barren plateaus during training.

The authors in [8,21] show that there are specific hierarchical architectures of hybrid quantum-classical models that are immune to barren plateaus. However, this massively restricts the available ansätze in a way that has not been explored yet.

But not only barren plateaus have an influence on the trainability of PQCs but also the composition of ansatz and handling of local minima. However, we do not want to focus on those criteria but want to evaluate the loss function. Here, we focus on a well-functioning method from metric learning, the Triplet Loss. Further we also care about noise robustness, that is the functioning with lower number of layers and small number of qubits as deep circuits are not useful in the NISQ era.

3 Metric Learning and Triplet Loss in Classical Machine Learning

The task of metric learning is closely related to ranking loss functions. In contrast to most loss functions where the output of a model is compared to a corresponding label or specific value, they predict relative distances between inputs, $x \in X$. The model is a function $f_\theta(x) : \mathbb{R}^D \mapsto \mathbb{R}^C$ that maps the D-dimensional input into a C-dimensional embedding space. During training, its parameters θ are updated according to the current loss value. The objective of the loss function is to ensure that semantically close inputs are also metrically close in the embedding space [13]. To do so, a similarity score between the points of the dataset has to be known. This can be realized for example based on class labels. The class labels can be interpreted as a binary score, i.e. the points belong to the same class or a different one, which is sufficient for many ranking loss functions [12].

This binary score is also utilized for Triplet Loss which was introduced by Schroff et al. [23]. In their work, the authors introduce Triplet Loss as a metric learning approach in the context of face recognition with a Deep Neural Network. In each training step i, a triplet of face images is drawn from the training set X. The triplets consist of a so-called *anchor* x_i^{a}, a *positive* x_i^{p}, and a *negative* x_i^{n}. Aim of the approach is that the image of the anchor x_i^a representing a specific person is closer to all images x^p of the same person than to images of another person x^n. Following the idea of metric learning, the loss function is:

$$\mathcal{L}_{\mathrm{TL}}(x_i^{\mathrm{a}}, x_i^{\mathrm{p}}, x_i^{\mathrm{n}}) = \max(\|f_\theta(x_i^{\mathrm{a}}) - f_\theta(x_i^{\mathrm{p}})\|^2 - \|f_\theta(x_i^{\mathrm{a}}) - f_\theta(x_i^{\mathrm{n}})\|^2 + \alpha, 0). \quad (1)$$

The relative distance between the embedding of the anchor and the positive should be close and the distance between the anchor and the negative should be large. The parameter α enforces a margin between the two pairs. When the distance between the two pairs is greater than α, the loss becomes zero in this step.

Schroff et al. [23] cut the error rate in comparison to the best published result by 30 % on two datasets achieving new state-of-the-art results. The comparison is especially impressive since their approach has a much better representational efficiency using an embedding dimension of only 128.

Deep Metric Learning has become a highly active field of research achieving state-of-the-art results on many datasets [12].

4 Training a Parameterized Quantum Circuit with Triplet Loss

Our approach aims to combine the idea of Triplet Loss with the training of a parameterized quantum circuit. In regular training with a Cross-Entropy Loss, PQCs operate in high-dimensional space that is reduced to a single value by the measurement and then compared to the corresponding label. By applying Triplet Loss, our circuit is less restricted and more easily able to leverage the high-dimensionality of Quantum Computing.

As shown in Fig. 1, we split our approach into two parts: i) the training of a model based on a PQC with Triplet Loss; and ii) the evaluation of the model with a classical linear support vector machine (SVM) [6]. In each training step, we run the PQC with the same parameters θ and encode three different inputs given a specific labeled input dataset: an anchor, a positive of the same class, and a negative of a different class. The PQC is used to create an embedding of each of the inputs. The dimensionality C of these embeddings is defined by the number of measurement operators, i.e. the number of measured qubits, of the circuit. The calculation of the Triplet Loss based on the three embeddings is unchanged in comparison to the classical approach, see eq. 1. The computation of gradients and update of circuit parameters are done with the parameter-shift rule [25]. Roughly speaking, this means the gradient of parameters $\nabla_\theta f_\theta(x)$ can be calculated from a finite parameter shift s as $\|f_{\theta-s}(x) - f_{\theta+s}(x)\|$. In effect, this allows us to derive the gradients of parameters of a parameterized quantum circuit using the same circuit.

To evaluate the model and perform classification on unknown data, we train a linear SVM with the embedded training data which we obtain by running each data point through the trained PQC. For evaluation of the test data, we use the same circuit pipeline to obtain the respective embeddings which are subsequently classified by the trained SVM.

Since current NISQ devices only have a limited amount of qubits, training on high-dimensional data is not possible. Therefore, one either has to use datasets with low dimensionality or perform a dimensionality reduction for the circuit's input. For the latter, we include a preprocessing step into our training and evaluation pipelines. This preprocessing step reduces the inputs to D-dimensional features to not exceed the number of available qubits and thus enables encoding them onto the circuit.

The model represented by the PQC can be expressed as follows:

$$f_\theta(q^D) = \langle 0^{\otimes n}|U^\dagger(q^D,\theta)M^{\otimes C}U(q^D,\theta)|0^{\otimes n}\rangle\,, \tag{2}$$

Here, $\langle 0^{\otimes n}|$ and $|0^{\otimes n}\rangle$ are the initial state of n qubits in the quantum computer. The operations to transform this initial state, according to our input vector q^D and the parameters θ to optimize, are expressed by the unitary operator $U(q^D,\theta)$ equivalent to the gates of our circuit. Finally, $M^{\otimes C}$ is a measurement M applied to C output qubits of interest. Notably, $C \leq n$ since some qubits are used for computations only.

5 Experiments

In the following, we present the setup of our experiments and the corresponding results. For a better understanding and reproducibility of our work, we first discuss the two datasets and the chosen constraints. We then analyze the accuracy of the trained models in light of our proposed approach. Finally, we examine the behavior of the gradients in the circuit. All our used datasets as well as the source code is publicly available for better reproducibility [29].

5.1 Datasets

For our work, we perform experiments on the moon dataset [1] provided by scikit-learn [20] and the well-known MNIST dataset [15] from classical ML. The moon dataset is a toy dataset of two interleaving half-circles on a two-dimensional plane. Each of the half-circles represents a different class and consists of 1250 data points in our experiments. Depending on a settable variable, noise can be added when the dataset is created to make the learning task more difficult. As the number of features of this dataset is two, no dimensionality reduction is required to encode the data onto the circuit. The only preprocessing step required is scaling the values to the range $[0, \pi]$.

The MNIST dataset consists of grayscaled images of handwritten digits with respective labels zero to nine. Each image has a size of 28×28 pixels and therefore is too large to be directly encoded on current QCs. Still, we chose to perform our experiments with this dataset as it is commonly used for various quantum classifiers and QML experiments. To reduce the number dimensionality of the dataset, we train a classical autoencoder. We prefer this preprocessing pipeline over using a principal component analysis (PCA) for dimensionality reduction as commonly done [26] as an autoencoder evenly distributes the sample information in the features. Our tests show that data generated by PCA is unbalanced leading to an easy classification task as the model can concentrate on a few or even one feature for good classification. After reducing the images, we scale the feature values to the range $[0, 2\pi]$ allowing better encoding.

Both datasets are divided into a training set and a test set. The training set consists of 2500 and the test set consists of 500 samples for both datasets. In both cases, the class labels are evenly distributed.

5.2 Parameter and Ansatz Selection

During the training phase (see Fig. 1), the triplets are chosen randomly from the training set. To make our experiments comparable, we run each experiment with $n = 12$ qubits. This is sufficient for up to 8 output qubits plus auxiliary qubits for complex computations. We chose to limit the number of features of the MNIST dataset to eight for every ansatz. We keep the Triplet Loss specific parameter α (see Eq. 1) at 1.0 since our experiments showed that different values appear to have no positive effect on our selected setup.

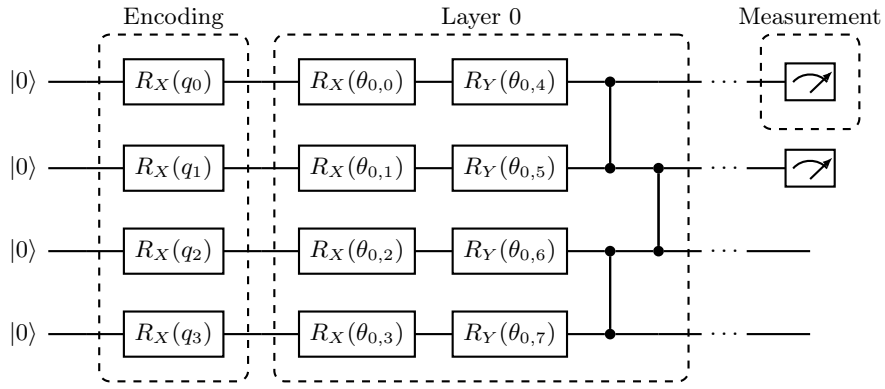


Fig. 3: Our chosen ansatz exemplarily visualized for four qubits and an embedding dimension of two. All qubits are initialized to the zero-state. The ansatz consists of the encoding part and one variational circuit layer with parameterized R_X and R_Y gates as well as entangling CZ gates. The variational layer can be repeated several times before measurements take place. Whithout any changes, this circuit can equally be used with Cross-Entropy Loss as well as Triplet Loss for a dataset with two classes.

As shown in Fig. 3, our ansatz is composed of a data encoding, a variational circuit as well as a measurement. For data encoding, we use angle encoding with R_X gates to encode the classical data. Following the data encoding, we employ a variational circuit with parameterized gates R_X and R_Y as well as non-parameterized entangling CZ gates. We use ten layers of the variational circuit during experiments and the analytical mode to get the exact expectation values of the measurements. Although our experiments show a better results for ZZ observables for measurements, we use a simple Z measurement as this allows for direct comparison with the associated ansatz for Cross-Entropy Loss. Furthermore, a simple Z measurement requires only one qubit while a ZZ measurement needs two qubits. This allows to perform extensive experiments with varying numbers of embedding dimensions. We conduct experiments for embedding dimensions from two to eight for up to three classes.

To evaluate the proposed Triplet Loss (TL) approach, we set up a second pipeline training a circuit commonly done in the literature and described in Sec. 2. So for each class, we measure one corresponding qubit and calculate a regular Cross-Entropy (CE) Loss in each training step:

$$\mathcal{L}_{\text{CE}}(x, y) = -\sum y * log(f_\theta(x)), \tag{3}$$

where x is the input, y the corresponding label and f_θ the model represented by the PQC. Therefore, this pipeline does not need a subsequent SVM to perform classification and evaluate the model as the highest expectation value is interpreted as the model's prediction. In contrast to our Triplet Loss approach, the output

Table 1: Average accuracy for varying embedding dimensions on the Moon and MNIST datasets for ten runs. For better comparibility, accuracies of TL are evaluated after 2000 training steps while 6000 training steps are considered for CE runs.

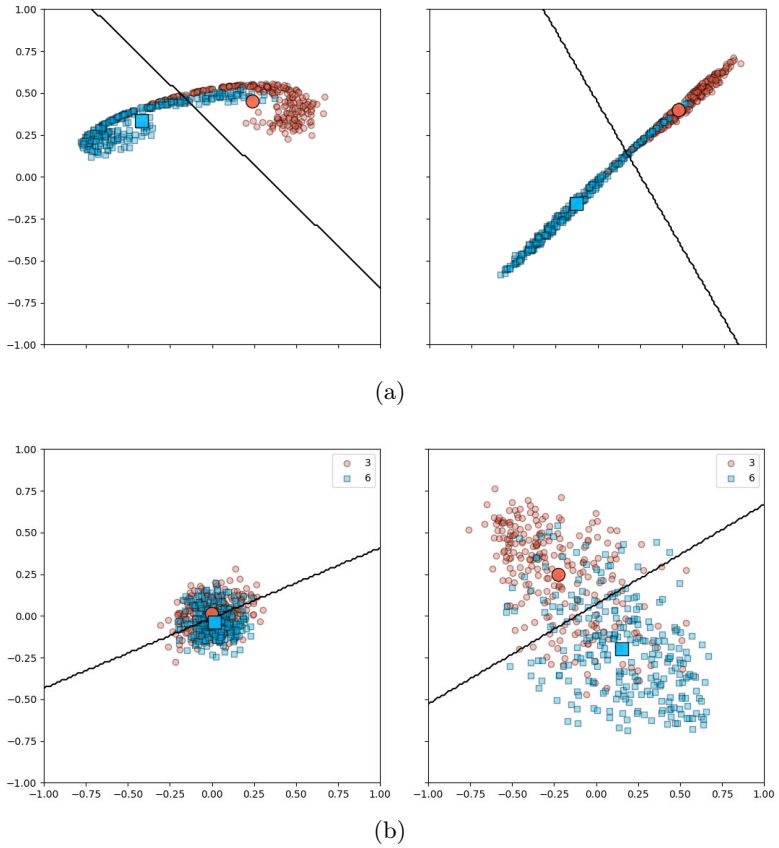
	TL - Embedding Dimension							
Classes	2	3	4	5	6	7	8	CE
Moon Dataset								
2	0.873	0.874	0.875	0.875	0.875	0.875	0.874	0.844
MMNIST Dataset								
2	0.775	0.788	0.803	0.817	0.843	0.843	0.853	0.800
3	0.786	0.801	0.834	0.853	0.866	0.873	0.884	0.767
4	0.578	0.612	0.651	0.695	0.735	0.755	0.754	0.620

dimension of the circuit is fixed by the number of classes. For the moon dataset, the number of classes is 2. For the MNIST dataset we conduct experiments for 2, 3, and 4 classes. To ensure comparability, the number of qubits, layers, and the ansatz is identical for TL and CE runs. Although the number of gradient updates is the same with 2000 training steps respectively, it is in the essence of Triplet Loss that the circuit has to be called three times as much. Therefore, we ensure that each CE run uses three times as many steps to make the results better comparable.

5.3 Results

Tab. 1 shows the accuracy for up to eight embedding dimensions averaged over ten runs for each experiment for the Moon and MNIST dataset. The Triplet Loss approach can always outperform the regular Cross-Entropy training given an appropriate number of embedding dimensions. In all cases, even for small embedding dimensions of no more than four, the Triplet Loss rivals the Cross-Entropy approach within a few percentage points. Additional embedding dimensions improve the performance of Triplet Loss consistently beyond Cross-Entropy.

For both datasets, a higher number of embedding dimensions for the Triplet Loss leads to better separation and thus better accuracy. However, this effect seems to stagnate in case of limited expressibility of the model. This can be seen for the MNIST dataset with four classes at seven or more embedding dimensions and the Moon dataset. The accuracy seems to converge, suggesting no further improvements with increasing number of embedding dimensions. However, a much larger number of experiments would be necessary to identify dependencies and regularities. But since a higher number of measurement operators would also lead to a higher number of qubits and thus larger circuits, trainings on current hardware and simulators would be computationally expensive and hard to compare.



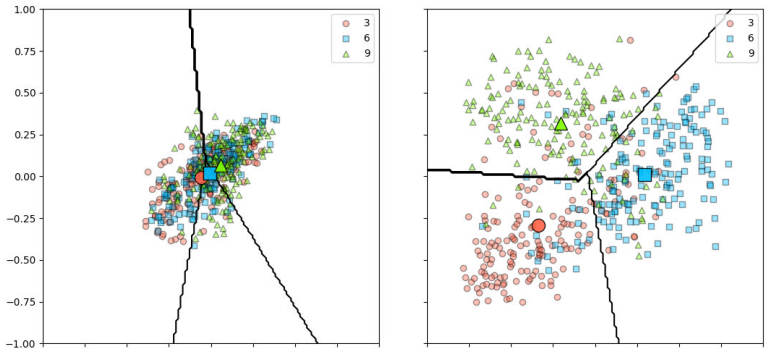
(a)

(b)

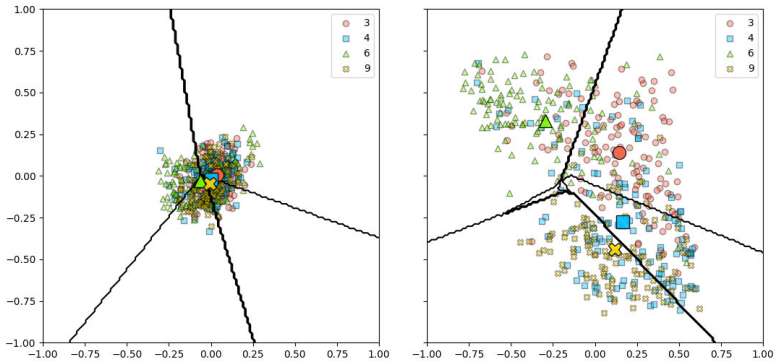
Fig. 4: Test set output after 50 (left) and 2000 (right) training steps for the moon dataset (a), and digits 3 and 6 of the MNIST dataset (b). The top and bottom as well as left and right figures share the same x- and y-axes. The bigger symbols represent the respective center of the clusters. The black line is the linear decision boundary determined by the SVM.

A drop in accuracies can be noted for the MNIST dataset when comparing the different classes grouped by their respective embedding dimensions. While the accuracies slightly improve from two and three classes, there is a decrease of the accuracy for four classes. This suggests that the expressibility of the model is not sufficient to learn an efficient embedding. Future experiments should consider increasing the number of layers.

As shown in Fig 4 for two classes on both datasets, the learning behavior corresponds to the intended one: In the course of the training, the two centers of the clusters are pushed further and further apart and the separation of the two classes becomes clearer. The two different runs also underline that Triplet Loss does not specify the location of the clusters and the mapping. While for the



(a)

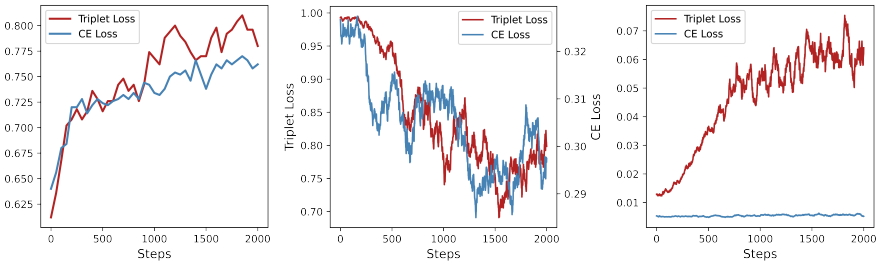


(b)

Fig. 5: Test set output after 50 (left) and 2000 (right) training steps for the MNIST dataset for the three digits 3, 6, and 9 (a) and the four digits 3, 4, 6, and 9 (b). The top and bottom as well as left and right figures share the same x- and y-axes. The bigger symbols represent the respective center of the clusters. The black line is the linear decision boundary determined by the SVM.

moon dataset shown in Fig 4a the whole test dataset is mapped on a straight line, the two digits of the MNIST dataset shown in Fig 4b form two separate round clusters. This behavior can also vary for comparable runs of the same dataset with a different parameter initialization.

Fig. 5 shows the SVM output after 50 and 2000 training steps. Fig. 5a and 5b confirm the numerical results for three and four classes respectively. While the separation of three classes is fairly distinct with good numerical accuracy, Triplet Loss cannot clearly separate four digits of the MNIST dataset. For four classes, three classes are separated well but the fourth one is spread across all regions. A modification of the original Triplet Loss equation to encourage a broader mapping might be a possible solution.



(a) Accuracy (b) Loss (c) Gradients

Fig. 6: Comparison of Triplet Loss (red) with a Cross-Entropy (blue) training regarding the accuracy, the loss curve and the average absolute gradient during 2000 steps.

During the training of classical models, the triplet selection has a direct influence on the performance and progression of the training and is one of the most studied questions in the field of metric learning [30,31]. Although this was not the focus of our work, we notice that a random selection seems to work best. Selecting hard triplets, where the negative is close to the anchor and the positive is further away, does not lead to any learning. But to further improve scores and derive conclusions, more experiments are needed.

Fig. 6 shows a comparison of various metrics during a training with Triplet Loss and with a Cross-Entropy Loss. In both experiments, the accuracy (see Fig. 6a) quickly reaches a very high value, which can hardly be improved in the course of the training. This similar behavior also becomes apparent in the two loss curves in 6b although both curves differ significantly in their absolute values which is due to the different calculation of the two losses. However, it can be seen that the Triplet Loss continues to show large fluctuations during training, whereas the Cross-Entropy Loss remains much more steady. An explanation for this is provided by looking at the average absolute gradients in the entire circuit, shown in Fig. 6c. Gradients for Triplet Loss are much higher and allow further adjustments of the weights during the entire training.

5.4 Gradients

To further investigate the behavior of the gradients, we undertake similar experiments to McClean et al. [17] in their work on barren plateaus. We therefore analyze the variance of the first parameter in the first layer across different runs for the two experiment setups. The resulting variances of gradients are shown in Fig. 7. The blue lines show the behavior of the Cross-Entropy based training while the red lines show the behavior for the Triplet Loss based training. As can be clearly seen from the blue lines, we can reproduce the behavior of exponentially vanishing gradients towards zero as a function of the number of qubits as reported

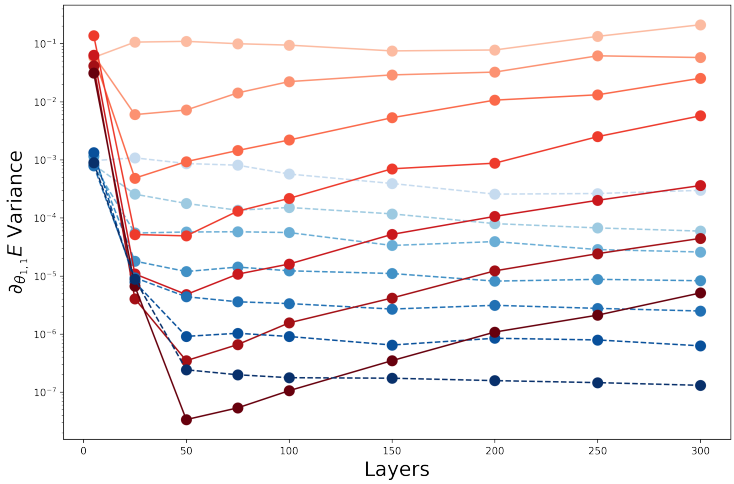


Fig. 7: Variance of the gradient for the first parameter over 100 runs. For the Triplet Loss approach (red) and the regular Cross-Entropy approach (blue). The different lines correspond to all even numbers of qubits between 4 and 16, starting from 4 quits at the top of the respective color panel.

by McClean et al. [17]. This implies that random initializations can cause the gradient-based optimization to fail.

For the Triplet Loss based training, a different behavior can be observed: While the gradient continues to decrease with an increasing number of qubits, it increases again with a higher number of layers. First, for qubit counts up to eight, the Triplet Loss based approach shows significantly higher variances of gradients. For ten qubits, the variances are comparable but improve significantly for layer counts bigger than 50. This can also be observed for bigger qubit counts. Thus, training with a higher number of layers may work with Triplet Loss, while a regular pipeline may not be able to learn because of effects of a barren plateau. This is also true for training circuits of arbitrary size on NISQ devices, where a small gradient is hardly distinguishable from noise. To investigate this further and to get insights on the scalability, a large number of computationally intensive simulations and extensive training on NISQ devices are required.

6 Conclusion

In this work, we have demonstrated the applicability of Triplet Loss for training a parameterized quantum circuit in a Quantum Machine Learning pipeline. For this purpose, we compared our pipeline in extensive experiments with a Cross-Entropy approach that is currently prominently used in literature. On two datasets, we were able to demonstrate competitive accuracies for both loss functions. As intended, using metric learning our models were able to consistently separate the

given classes. As our evaluation used a fixed circuit layout and thus computational power, for higher numbers of classes modifications of the Triplet Loss function and ansatz are conceivable to further improve the scores.

The observed high gradient values during training are particularly promising: These are significantly higher than those observed in conventional training on a QC. Still, owing to the current limitations of hardware and simulations our experiments were limited to a small number of layers. It remains a task for future work to explore scalability and generalizability on larger circuits. In the best case, Triplet Loss reduces the impact of barren plateaus and thus improves training in the area of QML. Especially in the NISQ era, a stronger gradient would improve the robustness against noise and thus increase the applicability of QML.

Acknowledgements The authors acknowledge support by the state of Baden-Württemberg through bwHPC.

References

1. Moon dataset. https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_moons.html, accessed: 2022-04-04
2. Benedetti, M., Lloyd, E., Sack, S., Fiorentini, M.: Parameterized quantum circuits as machine learning models. Quantum Science and Technology **4**(4), 043001 (nov 2019). https://doi.org/10.1088/2058-9565/ab4eb5
3. Bilkis, M., Cerezo, M., Verdon, G., Coles, P.J., Cincio, L.: A semi-agnostic ansatz with variable structure for quantum machine learning (2021), http://arxiv.org/abs/2103.06712
4. Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S.C., Endo, S., Fujii, K., McClean, J.R., Mitarai, K., Yuan, X., Cincio, L., Coles, P.J.: Variational quantum algorithms. Nature Reviews Physics **3**(9), 625–644 (aug 2021). https://doi.org/10.1038/s42254-021-00348-9
5. Cerezo, M., Sone, A., Volkoff, T., Cincio, L., Coles, P.J.: Cost function dependent barren plateaus in shallow parametrized quantum circuits. Nature Communications **12**(1) (mar 2021). https://doi.org/10.1038/s41467-021-21728-w
6. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning **20**(3), 273–297 (1995)
7. Dunjko, V., Wittek, P.: A non-review of quantum machine learning: trends and explorations. Quantum Views **4**, 32 (2020)
8. Grant, E., Benedetti, M., Cao, S., Hallam, A., Lockhart, J., Stojevic, V., Green, A.G., Severini, S.: Hierarchical quantum classifiers. npj Quantum Information **4**(1) (dec 2018). https://doi.org/10.1038/s41534-018-0116-9
9. Grant, E., Wossnig, L., Ostaszewski, M., Benedetti, M.: An initialization strategy for addressing barren plateaus in parametrized quantum circuits. Quantum **3**, 214 (Dec 2019). https://doi.org/10.22331/q-2019-12-09-214
10. Hettinger, C., Christensen, T., Ehlert, B., Humpherys, J., Jarvis, T., Wade, S.: Forward thinking: Building and training neural networks one layer at a time (2017)
11. Holmes, Z., Sharma, K., Cerezo, M., Coles, P.J.: Connecting ansatz expressibility to gradient magnitudes and barren plateaus. PRX Quantum 3, 010313, Published 24 January 2022 (2021). https://doi.org/10.1103/PRXQuantum.3.010313, http://arxiv.org/abs/2101.02138

12. Kaya, M., Bilge, H.Ş.: Deep metric learning: A survey. Symmetry **11**(9), 1066 (2019)
13. Kulis, B., et al.: Metric learning: A survey. Foundations and Trends® in Machine Learning **5**(4), 287–364 (2013)
14. LaRose, R., Coyle, B.: Robust data encodings for quantum classifiers. Phys. Rev. A 102, 032420 (2020) (2020). https://doi.org/10.1103/PhysRevA.102.032420, http://arxiv.org/abs/2003.01695
15. LeCun, Y., Cortes, C.: MNIST handwritten digit database (2010), http://yann.lecun.com/exdb/mnist/
16. Lloyd, S., Schuld, M., Ijaz, A., Izaac, J., Killoran, N.: Quantum embeddings for machine learning. arXiv preprint arXiv:2001.03622 (2020)
17. McClean, J.R., Boixo, S., Smelyanskiy, V.N., Babbush, R., Neven, H.: Barren plateaus in quantum neural network training landscapes. Nature communications **9**(1), 1–6 (2018)
18. McClean, J.R., Romero, J., Babbush, R., Aspuru-Guzik, A.: The theory of variational hybrid quantum-classical algorithms. New Journal of Physics **18**(2), 023023 (2016)
19. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press (2000)
20. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
21. Pesah, A., Cerezo, M., Wang, S., Volkoff, T., Sornborger, A.T., Coles, P.J.: Absence of barren plateaus in quantum convolutional neural networks. Phys. Rev. X **11**, 041011 (Oct 2021). https://doi.org/10.1103/PhysRevX.11.041011
22. Preskill, J.: Quantum computing in the nisq era and beyond. Quantum **2**, 79 (2018)
23. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 815–823 (2015)
24. Schuld, M.: Effect of data encoding on the expressive power of variational quantum-machine-learning models. Physical Review A **103**(3) (2021). https://doi.org/10.1103/PhysRevA.103.032430
25. Schuld, M., Bergholm, V., Gogolin, C., Izaac, J., Killoran, N.: Evaluating analytic gradients on quantum hardware. Physical Review A **99**(3), 032331 (2019)
26. Skolik, A., McClean, J.R., Mohseni, M., van der Smagt, P., Leib, M.: Layerwise learning for quantum neural networks. Quantum Machine Intelligence **3**(1), 1–11 (2021)
27. Thumwanit, N., Lortaraprasert, C., Yano, H., Raymond, R.: Trainable Discrete Feature Embeddings for Variational Quantum Classifier (2021), http://arxiv.org/abs/2106.09415
28. Wecker, D., Hastings, M.B., Troyer, M.: Progress towards practical quantum variational algorithms. Physical Review A **92**(4), 042303 (2015)
29. Wendenius, C., Kuehn, E.: quantum-triplet-loss (Jul 2022). https://doi.org/10.5281/zenodo.6786443
30. Xuan, H., Stylianou, A., Liu, X., Pless, R.: Hard negative examples are hard, but useful. In: European Conference on Computer Vision. pp. 126–142. Springer (2020)
31. Yu, B., Liu, T., Gong, M., Ding, C., Tao, D.: Correcting the triplet selection bias for triplet loss. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 71–87 (2018)